(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平10－63511

(43)公開日　平成10年(1998)３月６日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｇ０６Ｆ	9/45			Ｇ０６Ｆ	9/44	３２０Ｃ
Ｂ４１Ｊ	5/30			Ｂ４１Ｊ	5/30	Ｚ
	29/38				29/38	Ｚ
Ｇ０６Ｆ	3/12			Ｇ０６Ｆ	3/12	Ｃ
	9/445				9/06	５４０Ｇ

審査請求　未請求　請求項の数11　ＦＤ　（全 19 頁）　最終頁に続く

(21)出願番号　特願平8－231282

(22)出願日　平成８年(1996)８月14日

(71)出願人　000005496
富士ゼロックス株式会社
東京都港区赤坂二丁目17番22号

(72)発明者　阿部　仁
神奈川県足柄上郡中井町境430　グリーンテクなかい　富士ゼロックス株式会社内

(74)代理人　弁理士　岩上　昇一　（外１名）

(54)【発明の名称】　ジョブスクリプト実行装置

(57)【要約】

【課題】ジョブスクリプトが記述されたマシン上の実行環境と、そのジョブスクリプトを受け取り実行するマシン上の実行環境が異なった場合でも、ジョブスクリプトの実行を可能とすること。

【解決手段】ジョブスクリプトが記述されたマシン１１は、そのジョブスクリプトを受け取り実行するマシン１２に対してジョブスクリプトを渡す際に、マシン１２の実行環境を問い合わせて、マシン１２がジョブスクリプトを実行するのに必要な実行環境であるアプリケーションプログラムの機能あるいはプログラム全部をすべて備えているか否かを調べ、備えていない場合に、欠如している機能を補うための拡張機能モジュールないしはプログラム全部をジョブスクリプトに付加して送出する。マシン１２は、その拡張機能モジュールないしはプログラム全部が付加されたジョブスクリプトを受け取ると、拡張機能抽出手段によりそれらの付加情報を抽出し、抽出した拡張機能モジュールあるいはプログラムを実行環境とするよう実行環境の拡張を行う。この拡張された実行環境によりジョブスクリプトの実行が可能となる。

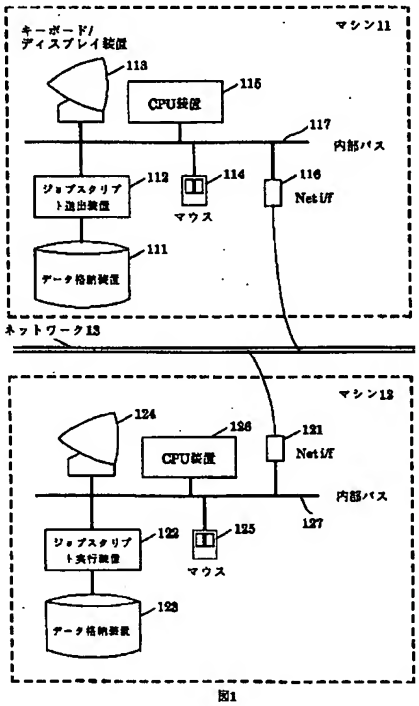

図1

1

【特許請求の範囲】

【請求項１】　任意の実行環境を有するジョブスクリプト実行手段と、

ジョブスクリプトの実行に必要な実行環境とそのジョブスクリプトをこれから実行させようとする前記ジョブスクリプト実行手段の実行環境の相違を判定する判定手段と、

その判定手段による判定の結果、相違があったときに、その相違を補うようにジョブスクリプト及び第２の実行環境のいずれか一方又は両方に変更を施す変更手段とを設けたことを特徴とするジョブスクリプト実行装置。

【請求項２】　第１の実行環境を持つ第１のジョブスクリプト実行装置から第２の実行環境を持つ第２のジョブスクリプト実行装置へ第１の実行環境を必要とするジョブスクリプトを送り出し、第２のジョブスクリプト実行装置は受け取ったジョブスクリプトを実行するジョブスクリプト実行システムにおいて、

第１のジョブスクリプト実行装置は、第２のジョブスクリプト実行装置の実行環境を問い合わせるための実行環境問合せ手段と、

その実行環境問合せ手段の問い合わせに応答して通知された第２の実行環境を第１の実行環境と比較するための実行環境比較手段と、

その実行環境比較手段による比較の結果、第２の実行環境が第１の実行環境を充足しない欠如部分を有するとき、その欠如部分を第２の実行環境に補足して第１の実行環境と等価な実行環境を生成するするための補足情報を、前記第２のジョブスクリプト実行装置に送出するジョブスクリプトに付加する付加手段とを有することを特徴とするジョブスクリプト実行装置。

【請求項３】　第２のジョブスクリプト実行装置は、

第１のジョブスクリプト実行装置からの実行環境の問い合わせがあった時に、それに応答する実行環境応答手段と、

受けとったジョブスクリプトから、それに付加された補足情報を抽出する抽出手段と、

前記抽出手段により抽出された補足情報を基に第２の実行環境を補足して、第１の実行環境と等価な実行環境を生成するための手段と、

アプリケーションプログラムを実行するアプリケーションプログラム実行手段とを有することを特徴とする請求項２記載のジョブスクリプト実行装置。

【請求項４】　コンピュータ上の複数のアプリケーションプログラムを逐次実行するジョブスクリプト実行装置が複数存在するシステムにおいて、

他のジョブスクリプト実行装置にジョブスクリプトを送出する際に、送付先のジョブスクリプト実行装置の実行環境を問い合わせる実行環境問合せ手段と、

前記実行環境問合せ手段の問い合わせの結果得られた実行環境のプログラムあるいはプログラムの機能とジョブスクリプトの実行に必要なプログラムあるいはプログラムの機能とを比較する実行環境比較手段と、

前記実行環境比較手段により比較した結果、不足するプログラムあるいはプログラムの機能を、ジョブスクリプトに付加する拡張機能付加手段とを設けたことを特徴とするジョブスクリプト実行装置。

【請求項５】　他のジョブスクリプト実行装置からの実行環境の問い合わせがあった時に、それに応答する実行環境応答手段と、

付加された拡張機能を実行するために、受けとったジョブスクリプトから付加された拡張機能を抽出する拡張機能抽出手段と、

前記拡張機能抽出手段により抽出された拡張機能をアプリケーションに付加して実行するアプリケーションプログラム実行手段とを設けたことを特徴とする請求項４記載のジョブスクリプト実行装置。

【請求項６】　ジョブスクリプトを読み込み解析するジョブスクリプト解析手段と、

類似の機能を持った複数のアプリケーションプログラムの機能対応表を備え、ジョブスクリプト解析手段の解析により特定された実行すべきアプリケーションプログラムが実行環境にないとき、前記アプリケーションプログラム機能対応表を用いて、必要な機能を持った代替のアプリケーションプログラムを選択するアプリケーションプログラム選択手段と、

選択されたアプリケーションプログラムを逐次実行するアプリケーションプログラム実行手段とを備えたことを特徴とするジョブスクリプト実行装置。

【請求項７】　必要な機能を持ったアプリケーションプログラムが存在しなかった場合に利用者にジョブスクリプトを実行するのに必要な当該機能がなかったことを知らせる警告手段を設けたことを特徴とする請求項６記載のジョブスクリプト実行装置。

【請求項８】　必要な機能を持ったアプリケーションプログラムが存在しなかった場合に、利用者の実行環境中に存在する複数のアプリケーションプログラムの機能の幾つかを組み合せることで当該アプリケーションプログラムの機能を構成し、前記アプリケーションプログラムの機能対応表に追加する機能構成追加手段を設けたことを特徴とする請求項６記載のジョブスクリプト実行装置。

【請求項９】　必要な機能を持ったアプリケーションプログラムが存在しなかった場合に、特定アプリケーションプログラムの特定機能や、利用者の実行環境中に存在する複数のアプリケーションプログラムを組み合せたもので、必要な当該アプリケーションプログラムの機能を代替しても良いと利用者が判断した場合に前記アプリケーションプログラムの機能対応表に代替機能を追加する代替機能追加手段を設けたことを特徴とする請求項６記載のジョブスクリプト実行装置。

【請求項１０】　ジョブスクリプト記述言語の種類を判定するジョブスクリプト記述言語判定手段と、
前記ジョブスクリプト記述言語判定手段により判定されたジョブスクリプト記述言語に対応してジョブスクリプトを解釈する複数のジョブスクリプト解釈手段と、
ジョブスクリプト解釈手段により解釈されたジョブスクリプトを特定のジョブスクリプト記述言語に翻訳し実行可能なジョブスクリプトとして出力するジョブスクリプト翻訳手段とを備えたことを特徴とするジョブスクリプト実行装置。

【請求項１１】　複数のジョブスクリプト実行装置が互いに通信可能な機能を有し、他のジョブスクリプト実行装置からの要求に従って、任意のジョブスクリプトを任意の形式のジョブスクリプトに翻訳するジョブスクリプト翻訳手段選択手段と、選択される複数のジョブスクリプト翻訳手段とを備え、変換したジョブスクリプトを要求元のジョブスクリプト実行装置に送出することを特徴とする請求項１０記載のジョブスクリプト実行装置。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、記述された順に、複数のアプリケーションプログラムを逐次実行するジョブスクリプト実行装置に関するものである。

【０００２】

【従来の技術】米国アップルコンピュータ社のＡｐｐｌｅＳｃｒｉｐｔ（商標）や、米国マイクロソフト社のＶｉｓｕａｌ　Ｂａｓｉｃ（商標）言語などの出現や、特開平２－３７４５４号公報に述べられている技術により、従来手作業で行っていた、キーボードからの同じ入力の繰り返しのような反復処理を自動化したり、複数の複雑なアプリケーションプログラムの機能を、あらかじめ決めた順序で順次実行させるなどの、プログラムのオートメーション化が可能になってきている。特にこれらの技術は、グラフィックユーザインタフェース（ＧＵＩ＝Ｇｒａｐｈｉｃａｌ　Ｕｓｅｒ　Ｉｎｔｅｒｆａｃｅ）を含めた操作の記録と再生、記録したジョブスクリプトの編集による自動化などが容易に行える点が、より以前からあるＵｎｉｘ上のＳｈｅｌｌ　Ｓｃｒｉｐｔなどと異なっている。

【０００３】特開平７－１２１３８２号公報では、例えばこのようなジョブスクリプトがいくつもの別の書式で書かれていた場合、１つの処理系でこれを解釈実行するための手段が記載されている。

【０００４】一方で、アドビシステムズ社のＰｈｏｔｏＳｈｏｐ（商標）やＱｕａｒｋ社のＱｕａｒｋＸｐｒｅｓｓ（商標）など幾つかのアプリケーションプログラムでは、アプリケーションプログラム自体の機能を拡張することができるように、プラグインやエクステンションモジュールというものを組み込むことができるようにするためのインタフェースを用意し、自社あるいはサードパーティによって容易に前記アプリケーションプログラムの機能を拡張する手段を提供している。

【０００５】

【発明が解決しようとする課題】しかし、上記ジョブスクリプトは、特定のアプリケーションプログラムを組み合わせた動作を記述するものであるため、ジョブスクリプトが記述されたマシン上の実行環境と、そのジョブスクリプトを受け取り実行するマシン上の実行環境が異なった場合や、アプリケーションプログラムがなかった場合、実行することができなかった。

【０００６】ジョブスクリプトではなく、ページ記述言語による例であるが、特開平５－１０８２８２号公報に、ページ記述言語にプログラム定義付加手段とプログラム解釈手段とを設け、特定のページ記述言語解釈プログラムが多様なページ記述言語による出力を可能にする技術が提案されている。この従来例と同様に、プログラム定義をジョブスクリプト自体に付加し、そのプログラム定義の解釈実行手段を設ければ、実行環境が異なっていても実行できる可能性はあるが、一般にジョブスクリプトは各種アプリケーションプログラムの外側で、それらを逐次実行することでプログラムのオートメーション化を試みるものであり、各アプリケーションプログラム自体の処理系は既存のものを利用しているだけである。このため、ジョブスクリプトの解釈実行手段にプログラム定義の解釈手段を設けることができず、異なった実行環境で作成されたジョブスクリプトは実行することができなかった。

【０００７】単にファイルをオープンするという場合に限って言えば、アップルコンピュータ社のパーソナルコンピュータでも、ファイルを生成したアプリケーションプログラムと同一のアプリケーションプログラムが動作マシン上になかった場合に、そのファイルを開くためのアプリケーションプログラムを利用者に問い合わせて指定させることにより、ファイルのオープンを可能にしているが、オープンした後にジョブスクリプトに記載されている各処理を指定されたアプリケーションプログラムによって実行させるための手段は提供されておらず、また２回目以降はファイルタイプにそのアプリケーションプログラムを対応づけてしまうため、逆に当該ファイルタイプを持つ２回目にオープンしたファイルが、対応づけられたアプリケーションプログラムに用意されていない機能を使用していた場合に、オープンできなくなるという問題点もあった。

【０００８】上記特開平７－１２１３８２号公報の従来技術では、スクリプト記述言語ごとに設けられた解釈手段で解釈した結果を、そのまま実行する、あるいはいったん中間言語にした後に実行するものであるので、スクリプト実行手段がスクリプト解釈手段の解釈結果を実行できる、あるいは中間言語を実行できるものである必要がある。ところが、一般的には各スクリプト記述言語

は、それに対応するスクリプト実行手段を備えた装置上で作成／記述されたものであり、実行手段自体は既存のものを利用する場合が多い。これは既存のものであるが故に、特開平７－１２１３８２号公報のようなスクリプト解釈手段の追加には対応しておらず、その結果スクリプト解釈手段の解釈結果や中間言語を実行するためには、新たにスクリプト実行手段を作る必要があった。また、上記特開平７－１２１３８２号公報のスクリプト実行手段は、複数のスクリプト記述言語に対応するものであるため、複数のスクリプト記述言語が必要とするすべての機能が実行可能なものでなければならず、その維持管理には、非常に多くのコストがかかるものであった。

【０００９】本発明は、ジョブスクリプトが記述されたマシン上の実行環境と、そのジョブスクリプトを受け取り実行するマシン上の実行環境が異なった場合、例えばアプリケーションプログラムがなかった場合や、記述言語が異なった場合でも、ジョブスクリプトの実行を可能とすることを課題とする。又、本発明は、特定のジョブスクリプト記述言語の実行環境しか持たないマシンからのジョブスクリプトの取得要求にも対応可能にすることを課題とする。

【００１０】

【課題を解決するための手段及び作用】本発明は、ジョブスクリプト実行装置において、任意の実行環境を有するジョブスクリプト実行手段と、ジョブスクリプトの実行に必要な実行環境とそのジョブスクリプトをこれから実行させようとする前記ジョブスクリプト実行手段の実行環境の相違を判定する判定手段と、その判定手段による判定の結果、相違があったときに、その相違を補うようにジョブスクリプト及び第２の実行環境のいずれか一方又は両方に変更を施す変更手段とを設けたことを特徴とする。あるジョブスクリプトを、それが作成された装置とは別の装置のジョブスクリプト手段で用いようとする場合、前述のように実行環境例えば実行に必要なアプリケーションプログラムやそのリソース、あるいはジョブスクリプト記述言語が異なることがあるので、そのままで実行することができるとは限らない。本発明では、ジョブスクリプトに必要な実行環境を、実行しようとしているジョブスクリプト実行手段の実行環境の相違を判定手段により判定する。その結果、実行環境に相違があれば、変更手段は、その相違により欠如する実行環境、例えばプログラムや拡張機能を付加して環境の補充により必要な環境を整えたり、あるいはジョブ記述言語が相違している例であればジョブスクリプトに変換を施す等の対処をする。これにより、異なった実行環境を備えた装置上で作成されたジョブスクリプトであっても、実行が可能となる。

【００１１】本発明（請求項２）は、第１の実行環境を持つ第１のジョブスクリプト実行装置から第２の実行環境を持つ第２のジョブスクリプト実行装置へ第１の実行環境を必要とするジョブスクリプトを送り出し、第２のジョブスクリプト実行装置は受け取ったジョブスクリプトを実行するジョブスクリプト実行システムにおいて、第１のジョブスクリプト実行装置は、第２のジョブスクリプト実行装置の実行環境を問い合わせるための実行環境問合せ手段と、その実行環境問合せ手段の問い合わせに応答して通知された第２の実行環境を第１の実行環境と比較するための実行環境比較手段と、その実行環境比較手段による比較の結果、第２の実行環境が第１の実行環境を充足しない欠如部分を有するとき、その欠如部分を第２の実行環境に補足して第１の実行環境と等価な実行環境を生成するするための補足情報を、前記第２のジョブスクリプト実行装置に送出するジョブスクリプトに付加する付加手段とを有する。又、本発明（請求項３）は、上記発明（請求項２）において、第２のジョブスクリプト実行装置は、第１のジョブスクリプト実行装置からの実行環境の問い合わせがあった時に、それに応答する実行環境応答手段と、受けとったジョブスクリプトから、それに付加された補足情報を抽出する抽出手段と、その抽出手段により抽出された補足情報を基に第２の実行環境を補足して、第１の実行環境と等価な実行環境を生成するための手段と、アプリケーションプログラムを実行するアプリケーションプログラム実行手段とを有する。

【００１２】これらの発明（請求項２、３）の作用においては、ジョブスクリプトが記述された第-1のジョブスクリプト実行装置から、第２のジョブスクリプト実行装置へジョブスクリプトを渡す際に、実行環境問合せ手段は、第２の実行装置の実行環境を問い合わせ、比較手段により第１の実行環境と第２の実行環境とを比較し、第２のジョブスクリプト実行装置がジョブスクリプトを実行するのに必要な実行環境をすべて備えていない場合に、その欠如している機能を補うための補足情報をジョブスクリプトに付加する。第２のジョブスクリプト実行装置は、その補足情報のあるジョブスクリプトを受け取ると、抽出手段により補足情報を抽出し、抽出した補足情報により実行環境の拡張を行う。このように、欠如した環境をジョブスクリプトに付加された補足情報に基づき補うことができるので、ジョブスクリプトが記述された第１のジョブスクリプト実行装置とジョブスクリプトを受け取り実行する第２のジョブスクリプト実行装置とで実行環境に相違があっても、ジョブスクリプトの実行が可能となる。

【００１３】前記発明（請求項２）の具体的な態様の発明（請求項４）は、コンピュータ上の複数のアプリケーションプログラムを逐次実行するジョブスクリプト実行装置が複数存在するシステムにおいて、他のジョブスクリプト実行装置にジョブスクリプトを送出する際に、送付先のジョブスクリプト実行装置の実行環境を問い合わせる実行環境問合せ手段と、その実行環境問合せ手段に

よる問い合わせの結果得られた実行環境のプログラムあるいはプログラムの機能とジョブスクリプトの実行に必要なプログラムあるいはプログラムの機能とを比較する実行環境比較手段と、その実行環境比較手段により比較した結果、不足するプログラムあるいはプログラムの機能を、ジョブスクリプトに付加する拡張機能付加手段とを設けたことを特徴とする。前記発明（請求項３）の具体的な態様の発明（請求項５）は、他のジョブスクリプト実行装置からの実行環境の問い合わせがあった時に、それに応答する実行環境応答手段と、付加された拡張機能を実行するために、受けとったジョブスクリプトから付加された拡張機能を抽出する拡張機能抽出手段と、前記拡張機能抽出手段により抽出された拡張機能をアプリケーションに付加して実行するアプリケーションプログラム実行手段とを設けたことを特徴とする。

【００１４】これらの発明（請求項４、５）の作用においては、ジョブスクリプトが記述されたマシンから、そのジョブスクリプトを受け取り実行するマシンに対してジョブスクリプトを渡す際に、ジョブスクリプトを受け取り実行するマシンの実行環境を問い合わせ、実行するマシンがジョブスクリプトを実行するのに必要な実行環境であるアプリケーションプログラムの機能あるいはプログラム全部をすべては備えていない場合に、欠如している機能を補うための拡張機能モジュールないしはプログラム全部をジョブスクリプトに付加する。ジョブスクリプトを受け取り実行するマシンは、その拡張機能モジュールないしはプログラム全部が付加されたジョブスクリプトを受け取ると、拡張機能抽出手段によりそれらの付加情報を抽出し、抽出した拡張機能モジュールによりアプリケーションプログラムの拡張あるいは付加されたプログラムを実行環境とする拡張を行う。このように、欠如した環境をジョブスクリプトに付加された情報に基づき補うことができるので、ジョブスクリプトが記述されたマシンとジョブスクリプトを受け取り実行するマシンとで実行環境に相違があっても、ジョブスクリプトの実行が可能となる。

【００１５】又、本発明（請求項６）は、ジョブスクリプトを読み込み解析するジョブスクリプト解析手段と、類似の機能を持った複数のアプリケーションプログラムの機能対応表を備え、ジョブスクリプト解析手段の解析により特定された実行すべきアプリケーションプログラムが実行環境にないとき、前記アプリケーションプログラム機能対応表を用いて、必要な機能を持った代替のアプリケーションプログラムを選択するアプリケーションプログラム選択手段と、選択されたアプリケーションプログラムを逐次実行するアプリケーションプログラム実行手段とを備えている。その作用においては、ジョブスクリプトを動作させるのに必要なアプリケーションプログラムが実行環境としてマシン上に存在しなかった場合には、アプリケーションプログラム選択手段は前記対応表を参照して適切な第２のアプリケーションプログラムを選択し、アプリケーションプログラム実行手段は、選択されたアプリケーションプログラムを実行する。したがって、ジョブスクリプトが記述されたマシン上の実行環境と、そのジョブスクリプトを受け取り実行するマシン上の実行環境が異なった場合でも、類似の機能を持つ別のアプリケーションプログラムが存在すれば当該ジョブスクリプトを実行できる。

【００１６】上記発明において、必要な機能を持ったアプリケーションプログラムが存在しなかった場合には、次のような対処方法がある。

（１）利用者にジョブスクリプトを実行するのに必要な当該機能がなかったことを知らせる警告手段を設ける（請求項７）。

【００１７】（２）利用者の実行環境中に存在する複数のアプリケーションプログラムの機能の幾つかを組み合せることで当該アプリケーションプログラムの機能を構成し、前記アプリケーションプログラムの機能対応表に追加する機能構成追加手段を設ける（請求項８）。

【００１８】（３）特定アプリケーションプログラムの特定機能や、利用者の実行環境中に存在する複数のアプリケーションプログラムを組み合せたもので、必要な当該アプリケーションプログラムの機能を代替しても良いと利用者が判断した場合に前記アプリケーションプログラムの機能対応表に代替機能を追加する代替機能追加手段を設ける（請求項９）。

【００１９】本発明（請求項１０）は、ジョブスクリプト記述言語の種類を判定するジョブスクリプト記述言語判定手段と、そのジョブスクリプト記述言語判定手段により判定されたジョブスクリプト記述言語に対応してジョブスクリプトを解釈する複数のジョブスクリプト解釈手段と、そのジョブスクリプト解釈手段により解釈されたジョブスクリプトを特定のジョブスクリプト記述言語に翻訳し実行可能なジョブスクリプトとして出力するジョブスクリプト翻訳手段とを備えたことを特徴とする。この構成により、動作環境が異なったジョブスクリプト記述言語で作成されたジョブスクリプトを動作させるのに、すべてのジョブスクリプト記述言語に対応するジョブスクリプト実行装置を作ることなしに、既存のジョブスクリプト実行手段を利用しながら、多言語によるジョブスクリプトを動作させることが可能になる。

【００２０】又、本発明（請求項１１）は、上記発明の複数のジョブスクリプト実行装置が互いに通信可能な機能を有し、他のジョブスクリプト実行装置からの要求に従って、任意のジョブスクリプトを任意の形式のジョブスクリプトに翻訳するジョブスクリプト翻訳手段選択手段と、選択される複数のジョブスクリプト翻訳手段とを備え、変換したジョブスクリプトを要求元のジョブスクリプト実行装置に送出する。複数のジョブスクリプト翻訳手段とそのジョブスクリプト翻訳手段選択手段を設け

たことにより、多言語のジョブスクリプトへの変換が可能となり、特定のジョブスクリプト記述言語の実行環境しか持たないマシンからのジョブスクリプトの取得要求にも対応が可能となる。

【００２１】

【発明の実施の形態】

（第１の実施形態）本発明における第１の実施の形態は、ジョブスクリプトが記述されたマシンから、そのジョブスクリプトを受け取り実行するマシンに対してジョブスクリプトを渡す際に、ジョブスクリプトを受け取り実行するマシンの実行環境を問い合わせ、実行するマシンがジョブスクリプトを実行するのに必要な実行環境をすべて備えていない場合に、欠如している機能を補うための拡張機能モジュールないしはプログラム全部をジョブスクリプトに付加することによりジョブスクリプトの実行を可能にしたものである。

【００２２】図１は本発明における第１の実施の形態によるジョブスクリプト実行装置の基本的なブロック図である。このジョブスクリプト実行装置は、マシン１１とマシン１２がネットワーク１３に接続され、互いに通信しあうことが可能となっている。マシン１１は、ジョブスクリプトやアプリケーションプログラム、各種リソースファイル等を格納するデータ格納装置１１１と、格納されたジョブスクリプトを読み込み必要なマシンに送付するためのジョブスクリプト送出装置１１２と、入出力装置としてのキーボード／ディスプレイ装置１１３及びマウス１１４と、それにＣＰＵ装置１１５が内部バス１１５上で結合されている。又、マシン１１は、ネットワークインタフェース１１６を有している。今、マシン１１はマシン１２にジョブスクリプトを送出しようとしているものとする。

【００２３】ネットワーク１３にネットワークインタフェース１２１を介してつながれたマシン１２は、受け取ったジョブスクリプトを実行するためのジョブスクリプト実行装置１２２と、マシン１１と同様にデータ格納装置１２３、キーボード／ディスプレイ装置１２４とマウス１２５、ＣＰＵ装置１２６などから成り、これらもまた内部バス１２７上で結合されている。

【００２４】ジョブスクリプト送出装置１１２は更に、図２に示すようにマシン１２に実行環境を問い合わせるための実行環境問合せ部２１と、問い合わせた実行環境をスクリプトで使用されているものと比較するための実行環境比較部２２と、比較した結果、何らかの機能拡張を行わないとマシン１２では当該ジョブスクリプトが実行できないことが判明した場合に、拡張機能をジョブスクリプトに付加するための拡張機能付加部２３から構成される。

【００２５】マシン１２側のジョブスクリプト実行装置１２２は、マシン１１からの実行環境の問い合わせがあった時に、それに答えるための実行環境応答部３１と、ジョブスクリプトに拡張機能が付加されてきた時にジョブスクリプトから拡張機能を抽出し、拡張機能とジョブスクリプトとを分離するための拡張機能抽出部３２と、抽出された拡張機能と共にジョブスクリプトの記述を逐次実行するアプリケーション実行部３３とを備えている。

【００２６】ここでのジョブスクリプトは、図４に示すように、複数のアプリケーションプログラムを参照／利用し、目的とする動作を実行する。さらに、各アプリケーションプログラムは、イメージファイルやテキストデータ、キー入力などのシステムの各リソースを使用しながら、各アプリケーションプログラムの目的とする動作を実行する。

【００２７】今、マシン１２に送出すべきジョブスクリプトは図６に例示するようなものであったとする。ここでは、「ＰｈｏｔｏＧｒａｐｈｅｒ」というアプリケーションが使われており、その中でファイルのオープン／クローズ、フィルタリングが行われている。この中で、オープン／クローズは、ほとんどのアプリケーションに共通しているものであり、「ＰｈｏｔｏＧｒａｐｈｅｒ」というアプリケーションが存在すれば必ず保持している機能だが、フィルタリングに関しての、ｆｉｌｔｅｒｌｉｎｇＦｉｌｅという機能はＰｌｕｇＩｎ機能によりＰｈｏｔｏＧｒａｐｈｅｒアプリケーションに追加していないと使用できない。

【００２８】そこでまず、図８の流れ図に従いマシン１１側では最初に当該ジョブスクリプトを読み込み（ステップ８－１）、ジョブスクリプトに使用されている機能を調査する（ステップ８－２）。ジョブスクリプトの読み込みと使用機能調査は、一般のコンパイラが備えるパーザに用いられる技術で実現可能であり、Ａｈｏ著の”Ｃｏｍｐｉｌｅｒｓ　Ｐｒｉｎｃｉｐｌｅｓ，　Ｔｅｃｈｎｉｑｕｅｓ，　ａｎｄ　Ｔｏｏｌｓ”等に記載されているもので、ここでは詳細については述べない。

【００２９】次にジョブスクリプトを送出するマシン１２の実行環境をジョブスクリプト送出装置中の実行環境問合せ部２１によって問い合わせる（ステップ８－３）。ステップ８－３では、マシン１２との通信を行い、その動作を簡単にマシン１２との間のプロトコルとして図９に示す。ここでは接続要求や問い合わせ要求が成功した場合にのみついて記載しているが、失敗した場合や応答がなかった場合の一定時間経過後のタイムアウトの処理については、単に通信失敗として終了するのみであるため、省略する。

【００３０】ここで問い合わせる実行環境とは、上記の図６のジョブスクリプトの例では、ＰｈｏｔｏＧｒａｐｈｅｒというアプリケーションがマシン１２にあるかどうか、またＰｈｏｔｏＧｒａｐｈｅｒアプリケーションのｆｉｌｔｅｒｌｉｎｇＦｉｌｅという拡張機能がマシン１２にあるかどうかを問い合わせるものであり、例え

ば図７に示すような使用アプリケーションごとの使用機能名のリストにヘッダーをつけた使用環境問合せファイルを送って、マシン１２からの応答を待つ。この問い合わせにはこの外にも例えば使用環境をリストにして送っても良い。

【００３１】そうして得たマシン１２の実行環境（ＰｈｏｔｏＧｒａｐｈｅｒというアプリケーションがあるか？、ｆｉｌｔｅｒｌｉｎｇＦｉｌｅ拡張機能があるか？）とジョブスクリプトで使用されている機能とを実行環境比較部２２によって比較し（ステップ８－４）、マシン１２側に拡張機能が必要でなければ（ステップ８－４）、そのままジョブスクリプトをマシン１２に送出して（ステップ８－７）終了する。しかし、もしもマシン１２側で拡張機能が必要であれば、ジョブスクリプトに、必要な拡張機能をマージ（ステップ８－６）した後にジョブスクリプトをマシン１２に送出して（ステップ８－７）終了する。ここで実行環境比較部２２は、単にステップ８－２で得た使用機能とステップ８－３の結果得られたマシン１２の実行環境とを１項目ずつ照合確認するのみである。

【００３２】また、ステップ８－６で拡張機能付加部２３は、マシン１２に対して送出するジョブスクリプトの先頭にジョブスクリプトの実行に用いられる拡張機能５２を、図５に示すように付加し、付加した拡張機能の数と、各々のサイズ等や置くべきディレクトリ（拡張機能フォルダーの名前等）を記載したヘッダー5-1と共に、ジョブスクリプト５３にマージする。そして、そのように作成した拡張機能の付加されたジョブスクリプトを送出する（ステップ８－７）。

【００３３】次に、マシン１２側の流れを図１０に従って説明する。まず、実行環境の問い合わせがマシン１１から来ると、実行環境応答部３１は、前記使用環境問合せファイルを読み込み、現在のマシン１２側の使用環境と比較する。ここで、マシン１２側の使用環境として要求されているアプリケーションが存在するかどうかは、マシン１２のコマンドサーチパス中や、ＯＳ自身の持つ検索機能（ＭａｃＯＳならファインダ等）を用いてマシン１２中を探索し、有無を調べる。また、拡張機能に関しても、拡張機能フォルダー中を含めた探索を行い、有無を調べ、その結果をマシン１１側に応答する（ステップ１０－１）。

【００３４】次に、マシン１１から送られてきたジョブスクリプトを受信し（ステップ１０－２）、ヘッダーの有無を調べる（ステップ１０－３）、拡張されたジョブスクリプトとしてのヘッダーがなければそのままジョブスクリプトを実行し（ステップ１０－６）、拡張されていれば、拡張機能抽出部３２によってジョブスクリプトから拡張機能とジョブスクリプト本体を分離する（ステップ１０－４）。分離に関しては、ヘッダーに記載されている拡張機能のサイズを見て、ファイルを分けるもので、分離そのものはＵｎｉｘオペレーティングシステムのｓｐｌｉｔコマンドなどに用いられている既存の技術である。そして、ヘッダーを参照して分離した拡張機能を必要な拡張機能フォルダーに追加し（ステップ１０－５）、アプリケーション実行部３３によりジョブスクリプトを実行する（ステップ１０－６）。

【００３５】ジョブスクリプトが記述されたマシン１１から、そのジョブスクリプトを受け取り実行するマシン１２に対してジョブスクリプトを渡す際に、マシン１２の実行環境を問い合わせ、そのマシン１２がジョブスクリプトを実行するのに必要な実行環境をすべて備えていない場合に、欠如している機能を補うための拡張機能モジュールないしはプログラム全部をジョブスクリプトに付加して送出する。マシン１２は、その付加情報のあるジョブスクリプトを受け取ると、その付加情報により実行環境の拡張を行う。このように、マシン１２に欠如した環境をジョブスクリプトに付加された情報に基づき補うことができるので、マシン１１とジョブスクリプトを受け取り実行するマシン１２とで実行環境に相違があっても、ジョブスクリプトの実行が可能となる。したがって、この実施形態により、異なった実行環境を備えたマシン上で作られたジョブスクリプトであっても、実行環境の違いを意識することなく、再実行させることが可能になる。

【００３６】（第２の実施形態）本発明の第２の実施形態は、ジョブスクリプトが記述されたマシン上の実行環境と、そのジョブスクリプトを受け取り実行するマシン上の実行環境が異なった場合でも、類似の機能を持つ別のアプリケーションプログラムが存在すれば当該ジョブスクリプトを実行できるように、アプリケーションプログラム名と機能名との対応表を設け、ジョブスクリプトを動作させるのに必要なアプリケーションプログラムが実行環境としてマシン上に存在しなかった場合に、前記対応表を参照して適切な第２のアプリケーションプログラムを起動させるアプリケーションプログラム選択部を設けたものである。図１１はその第２の実施の形態を構成する基本的なブロック図である。この装置は、ジョブスクリプトやアプリケーションプログラム、各種リソースファイル等を格納するデータ格納装置１１０１と、そのデータ格納装置１１０１に格納されたジョブスクリプトを読み込み実行するジョブスクリプト実行装置１１０２と、入出力装置としてのキーボード／ディスプレイ装置１１０３及びマウス１１０と、ＣＰＵ装置１１０５が内部バス１１０６上で結合された構成になっている。

【００３７】ジョブスクリプト実行装置１１０２は更に、図１２に示す様にジョブスクリプトファイルを読み込み解析するためのジョブスクリプト解析部１２０１と、本実施形態の中心であるアプリケーションプログラム選択部１２０２と、アプリケーションプログラム実行部１２０３からなっている。

【００３８】ジョブスクリプトは図４により前述したように、複数のアプリケーションプログラムを参照／利用し、目的とする動作を実行する。さらに、各アプリケーションプログラムは、イメージファイルやテキストデータ、キー入力などのシステムの各リソースを使用しながら、各アプリケーションプログラムの目的とする動作を実行する。

【００３９】図１３に、この構成でネットワークに接続された例を示す。今、ネットワークインタフェース１３０８を介して、外部から実行すべきジョブスクリプトと、テストファイル１が送られてきたものとする。ジョブスクリプト記述言語は、例えばアップルコンピュータ社のＡｐｐｌｅＳｃｒｉｐｔの例であり、そのジョブスクリプトの内容は、ここでは図１４に示すようなものであったと仮定する。このジョブスクリプトの場合、これを送ってきたマシン上には、クラリス社のファイルメーカーＰｒｏ（商標）というアプリケーションプログラムが存在していたため、これを実行することができたが、受け取った図１３で示すシステム側では、このアプリケーションプログラムが存在していないため、このままではこのジョブスクリプトを実行できない。

【００４０】図１６の流れ図に沿って説明すれば、図１３で示すシステムは、まずネットワークインタフェース１３０８を介して送られてきたジョブスクリプトＡを読み込み（ステップ１６－１）、ジョブスクリプト実行装置中のジョブスクリプト解析部1-2-0-1で解析される。ジョブスクリプト解析部１２０１は、一般のコンパイラが備えるパーザに用いられている技術と同じであり、Ａｈｏ著の”Ｃｏｍｐｉｌｅｒｓ　Ｐｒｉｎｃｉｐｌｅｓ，Ｔｅｃｈｎｉｑｕｅｓ，ａｎｄ　Ｔｏｏｌｓ”等に記載されているもので、ここでは詳細については述べない。次に図１６のステップ１６－２で、ジョブスクリプト実行装置中のアプリケーションプログラム選択部１２０２が、ジョブスクリプト中に記載されたアプリケーションプログラムの存在を確認する。具体的には、アップルコンピュータ社のＭａｃＯＳならばＦｉｎｄｅｒを用いてアプリケーションプログラムをファイルシステム中から探索し、ＵｎｉｘなどのＯＳならばコマンドサーチパス上のディレクトリ内を捜す。アプリケーションプログラムがシステム内にあれば、そのアプリケーションプログラムを実行する（ステップ１６－６）。

【００４１】またもしステップ１６－２で、ジョブスクリプト中に記載されたアプリケーションプログラムがシステム内に存在しなかった場合、対応表を参照して（ステップ１６－３）、必要とするアプリケーションプログラムと同等の機能を持ったアプリケーションプログラムが存在するかどうかを確認する（ステップ１６－４）。

【００４２】ここで参照されるアプリケーションプログラムと機能との対応表は、図１７に示すようなもので、各アプリケーションプログラムの機能が別のアプリケーションプログラム中に存在するかどうかの確認が可能である。ここでは、クラリス社のファイルメーカＰｒｏに対し、レコードの作成を要求しているが、この機能を持つアプリケーションプログラムを捜し、例えばマイクロソフト社のＥｘｃｅｌ（商標）というアプリケーションプログラムがこの機能を備えていたとすれば、ファイルメーカＰｒｏの代わりにＥｘｃｅｌでこの機能を実行させる。これは実際にジョブスクリプトを書き換えることはしないが、実行時には、仮想的に図１５のようなジョブスクリプトをジョブスクリプト解析部１２０１で解析したものと同じ実行イメージがメモリ上に展開され、ＣＰＵ装置（プロセッサ）１３０５によって実行される。

【００４３】また、高速化のために、例えば図１７のアプリケーションプログラム名／機能名対応表をアプリケーションプログラムの種類によって分類し、例えば図１８のようなドロー系ソフトの対応表１８１とペイント系ソフトの対応表１８２にしたものを用いることもできる。この場合、まず求めるアプリケーションプログラムがどのカテゴリーに属するアプリケーションプログラムであるかがわかっていることが前提であり、それはカテゴリー名とアプリケーションプログラム名との対応表とを持っても良いし、またもともとのジョブスクリプト中に、記載しておいても良い。

【００４４】ステップ１６－４の判定の結果、対応するアプリケーションプログラムが存在すれば、アプリケーションプログラム実行のステップ(ステップ1-6-6)に移り、終了する。

【００４５】もしも対応するアプリケーションプログラムが存在しなかった場合、当該ジョブスクリプトの実行ができないことを警告としてユーザに知らせて終了する。利用者に当該機能に対応するものがなかったことを知らせる警告部として、例えばスピーカの音声による警報とすることもできる。

【００４６】ここで、同等の機能を持った直接のアプリケーションプログラムの機能が存在しなかった場合に、ユーザへ警告する以外の手段で対処することもできる。その一例を以下に説明する。

【００４７】同等の機能を持った直接のアプリケーションプログラムの機能が存在しなかった場合でも、ユーザの判断で、別の機能を組み合せれば、当該機能と同等の機能を構成することができる場合、図１９の流れに従って、まずステップ１９－４で対応する同等の機能を持ったアプリケーションプログラム機能がないのでステップ１９－７を実行し、ここでユーザ定義可能かどうかをユーザに問い合わせる。もしもユーザ定義可能ならば、図２０のユーザ定義部２０－３によって、ユーザは、どれとどの機能を組み合せれば当該機能に相当する機能を構成できるかを指定し（ステップ１９－８）、それに従ってアプリケーションプログラムを起動する。もしもユーザ定義ができない場合は、そのまま終了となる。

【００４８】ここで作られるユーザ定義機能表は、例えば図２２で示すもので、機能１に対応する直接の機能がアプリケーションプログラム２になかった場合、アプリケーションプログラム２の機能２と機能５と機能６を組み合せれば、機能１と同等の効果が得られるということを利用者は指定し、図２２に示す表が作成される。このユーザ定義機能が作成された時点で、アプリケーションプログラム名／機能名対応表を、図２１に示すように、ユーザ定義機能の有無を保持するものする。図２２のユーザ定義機能表は、アプリケーションプログラムごとに持つものであるが、図２３のように、複数のアプリケーションプログラムの機能を組み合せられるような表もありうる。

【００４９】さらに図１９のステップ１９－７で別の機能を組み合せても当該機能と同等の機能を構成することができない場合でも、特定アプリケーションプログラムの特定機能や、利用者の実行環境中に存在する複数のアプリケーションプログラムを組み合せたもので、当該アプリケーションプログラムの機能を代替しても良いと利用者が判断する場合は、図２４の流れ図に示すように、ステップ２４－９からステップ２４－１０の代替機能定義に移り、利用者によって代替機能が指定され、実行される。ここで定義される代替機能の表は図２６および図２７に示すようなもので、データ構造としてはユーザ定義機能表とほぼ同じものになる。

【００５０】また、図２８はユーザ定義機能のユーザインタフェースの例で、図２０のユーザ定義部２０－３によって表示される。ここでは、メッセージによる対話形式のものとして表示しているが、アイコンを用いたグラフィカルユーザインタフェースであっても、もち論良い。この対話の結果として、ユーザ定義機能がユーザ定義機能表に書き込まれる。代替機能に関しても同様である。

【００５１】ここでユーザ定義機能と代替機能と比較すると、前者は機能的に同等な結果を与えるもので、一方後者は利用者にスクリプトの実行結果を想起させるためものであり、どんな表示でも構わない、例えば、丸い図形の代わりに「丸」と書かれた単なるボックスが配置されてもよい。

【００５２】この第２の実施形態によれば、ジョブスクリプトを動作させるのに必要なアプリケーションプログラムが実行環境としてマシン上に存在しなかった場合には、アプリケーションプログラム選択部２０－２は前記対応表を参照して適切な第２のアプリケーションプログラムを選択し、アプリケーションプログラム実行部２０－４は、選択されたアプリケーションプログラムを実行する従って、ジョブスクリプトが記述されたマシン上の実行環境と、そのジョブスクリプトを受け取り実行するマシン上の実行環境が異なった場合でも、類似の機能を持つ別のアプリケーションプログラムが存在すれば当該ジョブスクリプトを実行できる。

【００５３】（第３の実施形態）

第３の実施形態の第１の例

第３の実施形態の第１の例（実施例）は、ジョブスクリプト実行装置自体は既存のものを利用しながら、複数のジョブスクリプト言語に対応できるようにしたものである。これは、第２の実施形態の図１１に示す基本構成と同じ基本構成を有するので、以下においては図１１を参照して説明する。異なるのは、ジョブスクリプト実行装置１１０２を図２９に示す構成とした点である。ジョブスクリプト実行装置は、図２９に示す様にジョブスクリプトファイルを読み込み、どの種類のジョブスクリプト記述言語であるかを判定するジョブスクリプト記述言語判定装置２９０１と、判定された各ジョブスクリプト記述言語に対応したジョブスクリプト言語解釈装置２９０２ａ～２９０２ｎと、解釈されたジョブスクリプトを、特定のジョブスクリプト実行装置で動作させることができるように、特定のジョブスクリプト記述言語に翻訳するジョブスクリプト翻訳装置２９０３とを備え、それが、既存のジョブスクリプト実行部２９０４に入力され実行される構成のものである。

【００５４】今、図４のジョブスクリプト１がデータ格納装置１１０１に格納された図３０のようなジョブスクリプトＡであるとする。利用者からのスクリプト実行要求が、キーボード／ディスプレイ装置１１０３とマウス１１０４を介して行われると、データ格納装置１１０１内のジョブスクリプトＡが、ジョブスクリプト実行装置１１０２に渡され実行を促される。

【００５５】するとジョブスクリプト実行装置１１０２内のジョブスクリプト記述言語判定装置は、この図３０のジョブスクリプトＡをスキャンし、特徴的な構文「ｔｅｌｌ　ａｐｐｌｉｃａｔｉｏｎ」という記述を読み読み取ると、図３１の構文／スクリプト言語対応表を参照して、このスクリプトがＡｐｐｌｅＳｃｒｉｐｔで書かれているものであると判定し、ジョブスクリプト実行装置１１０２内の複数のジョブスクリプト言語解釈装置２９０２ａ～２９０２ｎから、ＡｐｐｌｅＳｃｒｉｐｔ用ジョブスクリプト言語解釈装置（２）２９０２ｂを選択する。選択されたＡｐｐｌｅＳｃｒｉｐｔ用ジョブスクリプト言語解釈装置（２）２９０２ｂは読み込まれたジョブスクリプトＡを解釈していく。ここでのジョブスクリプト言語の解釈は、一般のコンパイラが備えるパーザに用いられる技術で実現可能であり、Ａｈｏ著の”Ｃｏｍｐｉｌｅｒｓ　Ｐｒｉｎｃｉｐｌｅｓ，　Ｔｅｃｈｎｉｑｕｅｓ，　ａｎｄ　Ｔｏｏｌｓ”等に記載されているもので、ここでは詳細については述べない。

【００５６】次に、このジョブスクリプトＡを実行すべきマシン上には、Ｖｉｓｕａｌ　Ｂａｓｉｃの実行環境しかなかったとすれば、このジョブスクリプト実行装置１１０２内のジョブスクリプト翻訳装置２９０３は、前

記解釈されたスクリプトをＶｉｓｕａｌ　Ｂａｓｉｃの構文に逐次翻訳していく。ここでは、例えば図３２のように、入力されたジョブスクリプト記述言語の解釈結果と、対応するＶｉｓｕａｌ　Ｂａｓｉｃでの命令文の対応表を用いて考えるが、もう少し複雑な文法を持ったジョブスクリプト記述言語でも良く、その場合には目的とするジョブスクリプト実行装置に対応したジョブスクリプト記述言語に翻訳するために、評価順序の入れ替え、ループの展開などいろいろな目的ジョブスクリプト生成技術が必要になってくるが、本発明の実施に必ずしも必要であるものではないため、ここでは最も単純に対応表で変換する例のみを記述した。

【００５７】ここでは、ＭｉｃｒｏＳｏｆｔ　Ｅｘｃｅｌを扱ったＡｐｐｌｅＳｃｒｉｐｔの例になっているが、変換されたジョブスクリプトはＭｉｃｒｏＳｏｆｔ　ＥｘｃｅｌのＶＢＡとして表現されるため、ＭｉｃｒｏＳｏｆｔ　Ｅｘｃｅｌを起動する部分の記述はなく、代わりにサブルーチン名としてｄｕｍｍｙ１を与えている。図３３は、この様にして図３０のジョブスクリプトＡを変換して得られたＶｉｓｕａｌＢａｓｉｃのジョブスクリプトＢを示すものである。

【００５８】得られたＶｉｓｕａｌＢａｓｉｃのジョブスクリプトを既存のジョブスクリプト実行部２９０４（ＶｉｓｕａｌＢａｓｉｃの実行系）に与えて実行させる。

【００５９】以上に説明した第３の実施形態の第１の例によれば、動作環境が異なったジョブスクリプト記述言語で作成されたジョブスクリプトを動作させるのに、従来技術（特開平７－１２１３８２号公報）のようにすべてのジョブスクリプト記述言語に対応するジョブスクリプト実行装置を作ることなしに、既存のジョブスクリプト実行手段を利用しながら、多言語によるジョブスクリプトを動作させることが可能になるので、利用者のジョブスクリプト資産を有効に活用することができ、またその維持管理も低廉に行うことができる。

【００６０】第３の実施形態の第２の例

この第３の実施形態の第２の例は、複数のジョブスクリプト実行装置が、互いに通信可能出会った場合に、あるジョブスクリプト実行装置に対して、他のジョブスクリプト実行装置からジョブスクリプトの取得要求があった際に、取得要求もとのジョブスクリプト実行部で実行可能なジョブスクリプト記述言語に翻訳することができるようにしたものである。図３４は第３実施形態の第２の例によるジョブスクリプト実行装置を有するシステムのブロック図である。このシステムは、複数のジョブスクリプト実行装置が互いに通信可能に構成されている。すなわち、図１１と同様の構成のマシン３４１が、ネットワークインタフェース３４１６、３４２６を介して、ネットワーク３４３に接続され、マシン３４２と互いに通信しあうことが可能となっている。

【００６１】ジョブスクリプト実行装置は、図３５に示すように、ジョブスクリプトファイルを読み込み、どの種類のジョブスクリプト記述言語であるかを判定するジョブスクリプト記述言語判定装置３５０１と、判定された各ジョブスクリプト記述言語に対応したジョブスクリプト言語解釈装置３５０２ａ～３５０２ｎと、解釈されたジョブスクリプトを、複数種のジョブスクリプト実行装置のジョブスクリプト記述言語に翻訳する複数のジョブスクリプト言語翻訳装置３５０４ａ～３５０４ｎと、ジョブスクリプト言語解釈装置３５０２ａ～３５０２ｎのいずれか１つの出力を指定された複数種のジョブスクリプト言語のうちの指定された１つにより動作させることができるように、ジョブスクリプト翻訳装置選択装置３５０３により選択された言語解釈装置の解釈出力を指定された特定のジョブスクリプト言語解釈装置に選択的に与えるジョブスクリプト翻訳選択装置３５０３とを備えている

【００６２】今、マシン３４２からマシン３４１にジョブスクリプトの取得要求があったものとする。ここで、マシン３４１自体にはＡｐｐｌｅＳｃｒｉｐｔの実行環境が整っているが、マシン３４２にはＶｉｓｕａｌＢａｓｉｃの実行環境しかなかった場合、マシン３４２からは実行環境としてＶｉｓｕａｌＢａｓｉｃで実行可能なジョブスクリプトの取得要求が行われる。

【００６３】マシン３４１側では、これを受けて、要求のあったジョブスクリプトをジョブスクリプト実行装置３４１２を通して第１の例と同様に解釈し、図３５に示す第２の例のジョブスクリプト実行装置構成図中のジョブスクリプト翻訳装置選択装置３５０３によって、各ジョブスクリプト記述言語に対応した複数のジョブスクリプト言語翻訳装置３５０４ａ～３５０４ｎから、ＶｉｓｕａｌＢａｓｉｃに対応したジョブスクリプト翻訳装置（３）３５０４ｃを選択する。その結果として、第１の例と同様にＡｐｐｌｅＳｃｒｉｐｔからＶｉｓｕａｌＢａｓｉｃに翻訳されたジョブスクリプトは、ジョブスクリプト送出装置３５０３によって、マシン２に送出される。マシン３４２側では、そのようにして送られてきたジョブスクリプトを自マシンで実行し、処理を終える。このマシン１とマシン２とのやりとりは、図３６に示す。

【００６４】以上に説明した第３の実施形態の第２の例によれば、第１の例と同様に動作環境が異なったジョブスクリプト記述言語で作成されたジョブスクリプトを動作させるのに、従来技術のようにすべてのジョブスクリプト記述言語に対応するジョブスクリプト実行装置を作ることなしに、既存のジョブスクリプト実行手段を利用しながら、多言語によるジョブスクリプトを動作させることが可能になり、利用者のジョブスクリプト資産を有効に活用することができると共に、さらにジョブ言語解釈装置の出力を多言語のジョブスクリプトへ変換するた

めの選択可能な複数のジョブスクリプト言語翻訳装置を設けたので、特定のジョブスクリプト記述言語の実行環境しか持たないマシンからのジョブスクリプトの取得要求にも対応が可能となる。

【００６５】

【発明の効果】本発明（請求項１～請求項１１）によれば、ジョブスクリプトが記述されたマシン上の実行環境と、そのジョブスクリプトを受け取り実行するマシン上の実行環境が異なった場合でも、ジョブスクリプトの実行が可能となるので、利用者のジョブスクリプト資産を有効に活用することができる。

【００６６】又、本発明（請求項２～請求項５）によれば、ジョブスクリプトが記述されたマシンとジョブスクリプトを受け取り実行するマシンとで実行環境に相違があっても、その相違を補うための情報をジョブスクリプトに付加し、その付加情報を基に実行環境の拡張を行うことができるので、ジョブスクリプトの実行が可能となる。

【００６７】本発明（請求項６～請求項９）によれば、ジョブスクリプトを動作させるのに必要なアプリケーションプログラムが実行環境としてマシン上に存在しなかった場合には、アプリケーションプログラム選択手段は機能対応表を参照して適切な第２のアプリケーションプログラムを選択し、アプリケーションプログラム実行手段は、選択されたアプリケーションプログラムを実行するので、類似の機能を持つ別のアプリケーションプログラムが存在すれば当該ジョブスクリプトを実行することができる。必要な機能を持ったアプリケーションプログラムが存在しなかった場合には、（１）警告手段により利用者に知らせて（請求項７）適切な対処を促すことができ、あるいは、（２）前記機能構成追加手段（請求項８）により前記機能対応表を複数プログラムの機能の組み合わせをも含むよう拡張することにより、実行環境を拡張し、あるいは、（３）代替機能追加手段（請求項９）により、利用者の判断をも参照して更に一層の実行環境の拡張を図ることができる。

【００６８】本発明（請求項１０～請求項１１）によれば、動作環境が異なったジョブスクリプト記述言語で作成されたジョブスクリプトを動作させるのに、すべてのジョブスクリプト記述言語に対応するジョブスクリプト実行装置を作ることなしに、既存のジョブスクリプト実行手段を利用しながら、多言語によるジョブスクリプトを動作させることが可能になり、利用者のジョブスクリプト資産を有効に活用することができる。さらに、本発明（請求項１１）によれば、多言語のジョブスクリプトへの変換機構として複数の翻訳装置とその選択装置とを設けるので、特定のジョブスクリプト記述言語の実行環境しか持たないマシンからのジョブスクリプトの取得要求にも対応が可能となる。

【図面の簡単な説明】

【図１】　本発明の第１の実施形態のブロック図

【図２】　第１の実施形態におけるジョブスクリプト送出装置の構成を示す図

【図３】　第１の実施形態におけるジョブスクリプト実行装置の構成を示す図

【図４】　データ格納装置内の各データ及びその参照関係を示す図

【図５】　送出される拡張機能の付加されたジョブスクリプトの構成例を示す図

【図６】　実行すべきジョブスクリプトの例を示す図

【図７】　使用環境問合せファイルの構成例を示す図

【図８】　マシン１１側の処理の流れ図

【図９】　実行環境問合せ部によるマシン１１とマシン１２との通信プロトコルの例を示す図

【図１０】　マシン１２側の処理の流れ図

【図１１】　本発明の第２の実施形態のブロック図

【図１２】　ジョブスクリプト実行部の構成を示す図

【図１３】　ネットワークに接続される構成例を示す図

【図１４】　実行すべきジョブスクリプトの例を示す図

【図１５】　実際に処理される図１４に示すジョブスクリプトと同等のジョブスクリプト

【図１６】　第２の実施形態の動作を示す流れ図

【図１７】　アプリケーションプログラム名／機能名対応表を示す図

【図１８】　カテゴリー別アプリケーションプログラム名／機能名対応表を示す図

【図１９】　ユーザ定義によって代替機能の追加を行う例の流れ図

【図２０】　ジョブスクリプト実行装置の構成例を示す図

【図２１】　アプリケーション名／機能名対応表の例を示す図

【図２２】　ユーザ定義機能表の例を示す図

【図２３】　ユーザ定義機能表の田の例を示す図

【図２４】　ユーザ定義によって代替機能の追加ができない場合において、代替機能定義によって代替定義を行う例の流れ図

【図２５】　アプリケーション名／機能名対応表の他の例を示す図

【図２６】　代替機能表の例を示す図

【図２７】　代替機能表の他の例を示す図

【図２８】　ユーザ定義機能のユーザインタフェースの例を示す図

【図２９】　第３の実施形態のジョブスクリプト実行装置の構成例を示す図

【図３０】　実行すべきジョブスクリプトの例を示す図

【図３１】　ジョブスクリプト記述言語判定装置で参照される構文／スクリプト言語対応表の例を示す図

【図３２】　ジョブスクリプト翻訳装置で参照される意味／命令文対応表の例を示す図

【図３３】　変換後のＶｉｓｕａｌＢａｓｉｃのジョブスクリプトの例を示す図

【図３４】　第３の実施形態のジョブスクリプト実行装置の他の構成例を示す図

【図３５】　図３４におけるスクリプト実行装置の構成例を示す図

【図３６】　マシン３４１とマシン３４２との通信プロトコルの例を示す図

【符号の説明】

１１１、１２３…データ格納装置　　１１２…ジョブスクリプト送出部　　１１３、１２４…キーボード／ディスプレイ装置　　１１４、１２５… マウス１１５、１２６…ＣＰＵ装置（プロセッサ）　　１１６、１２１…ネットワークインタフェース　　１１７、１２７…内部バス　　１３…ネットワーク　　１２２…ジョブスクリプト実行装置

【図１】

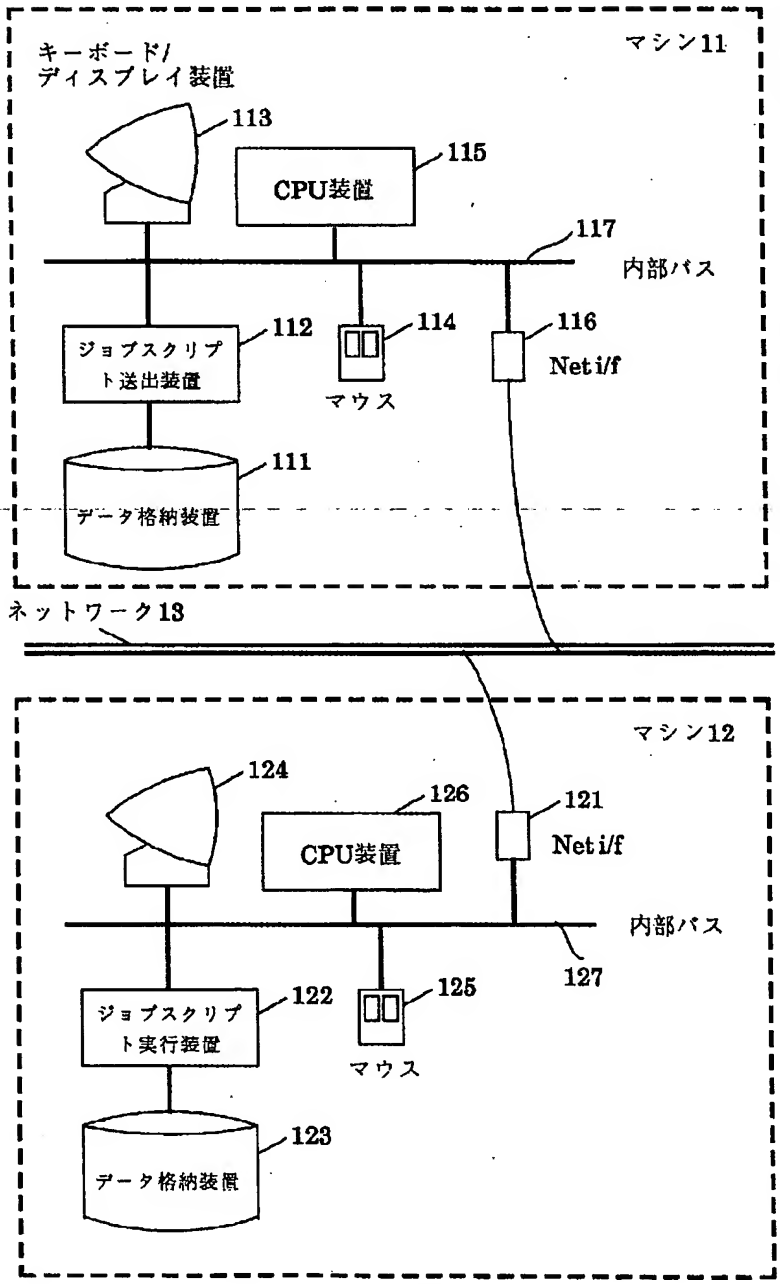

図1

【図６】

```
tell application "PhotoGrapher"
    Open file "テストファイル1"
    err=filteringFile {"テストファイル1"}
    Close file
end tell
```

図6: 実行すべきジョブスクリプト

【図７】

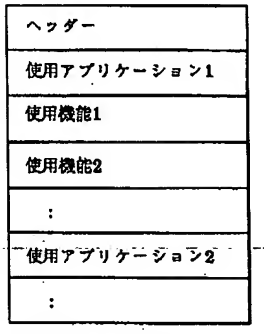

図7: 使用環境問合せファイル

【図１２】

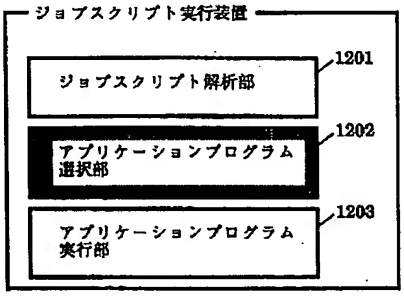

図12

【図２】

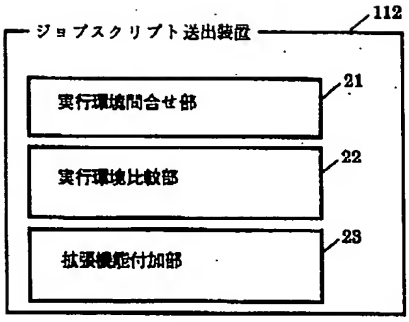

図2

【図３】

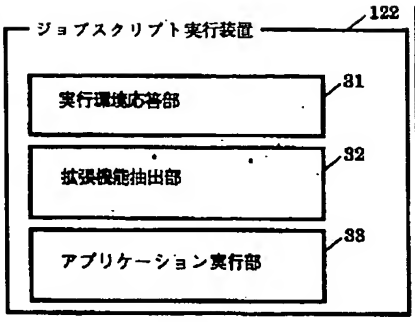

図3

【図２２】

欠如機能名	機能の組合せ
機能1	機能2＋機能5＋機能6
機能2	機能1＋機能3

図22: ユーザ定義機能表(1)

【図４】

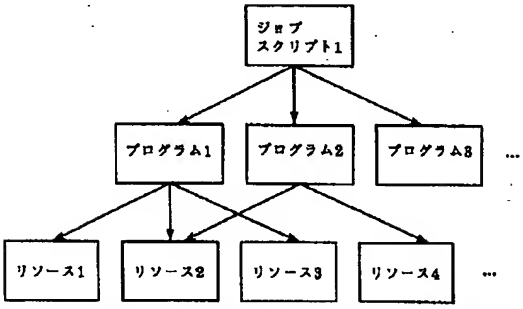

図4: データ格納装置内の各データ及びその参照関係

【図５】

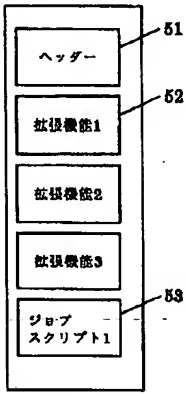

図5: 送出される拡張機能の付加されたジョブスクリプト

【図１１】

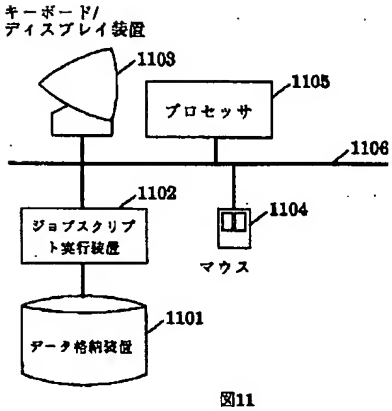

図11

【図１３】

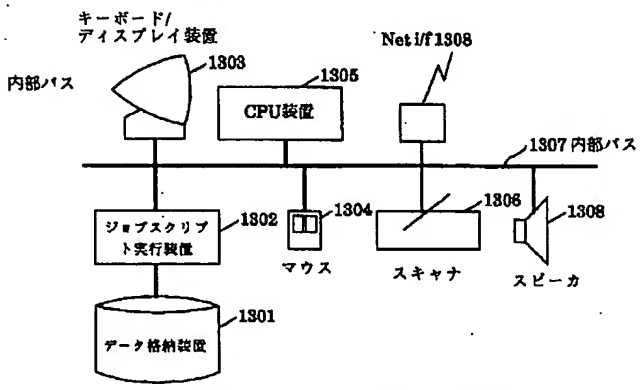

図13: ネットワークに接続された実施例

【図１４】

```
tell application "ファイルメーカーPro"
    Open file "テストファイル1"
    Create New Record with Data {"テストファイル1"}
    Close file
end tell
```

図14: 実行すべきジョブスクリプトA

【図８】

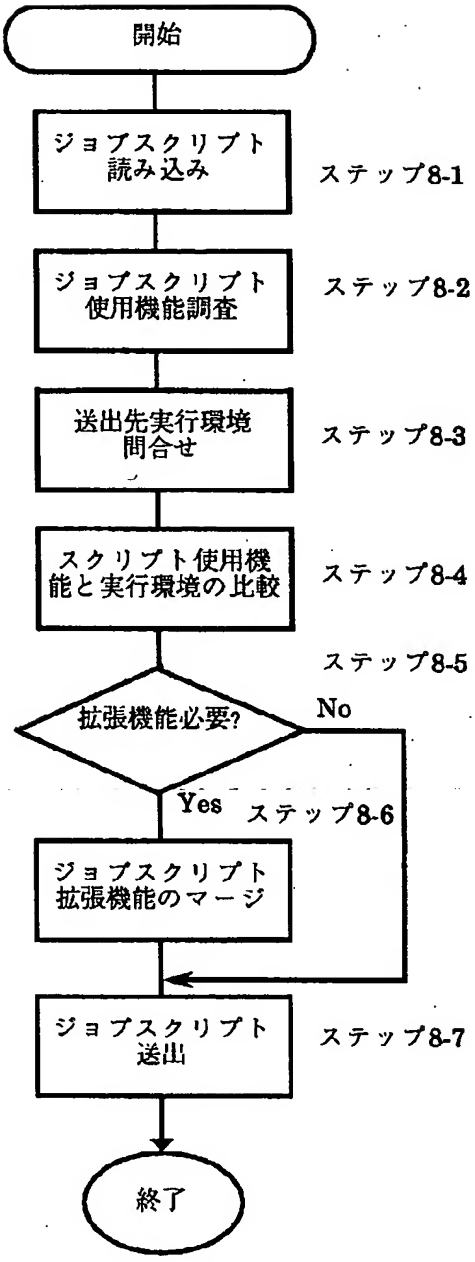

図8: マシン11側の処理の流れ

【図９】

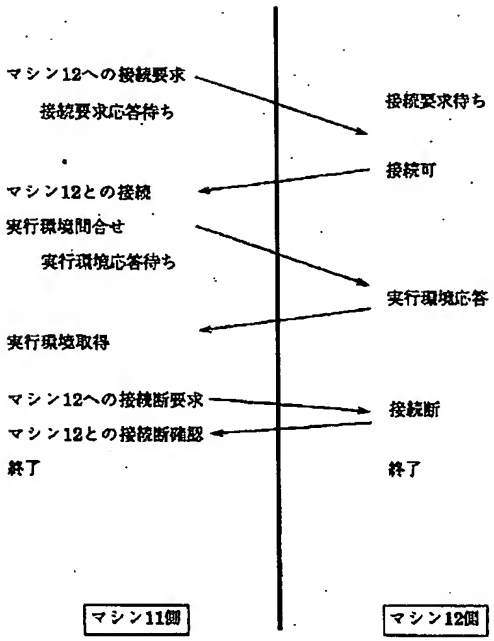

図9: 実行環境問合せ手段によって行われる
マシン11とマシン12との通信プロトコル

【図１５】

```
tell application "MicroSoft Excel"
    Open file "テストファイル1"
    Create New Record with Data {"テストファイル1"}
    Close file
end tell
```

図15: 実際に処理されるものと同等のジョブスクリプトB

【図１６】

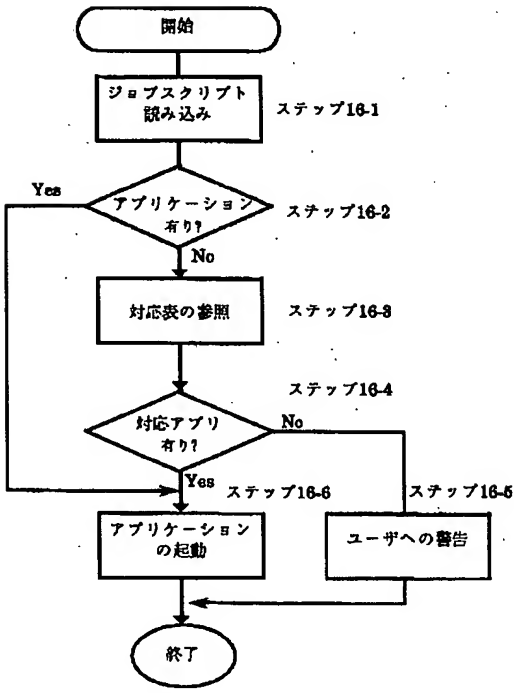

図16

【図１０】

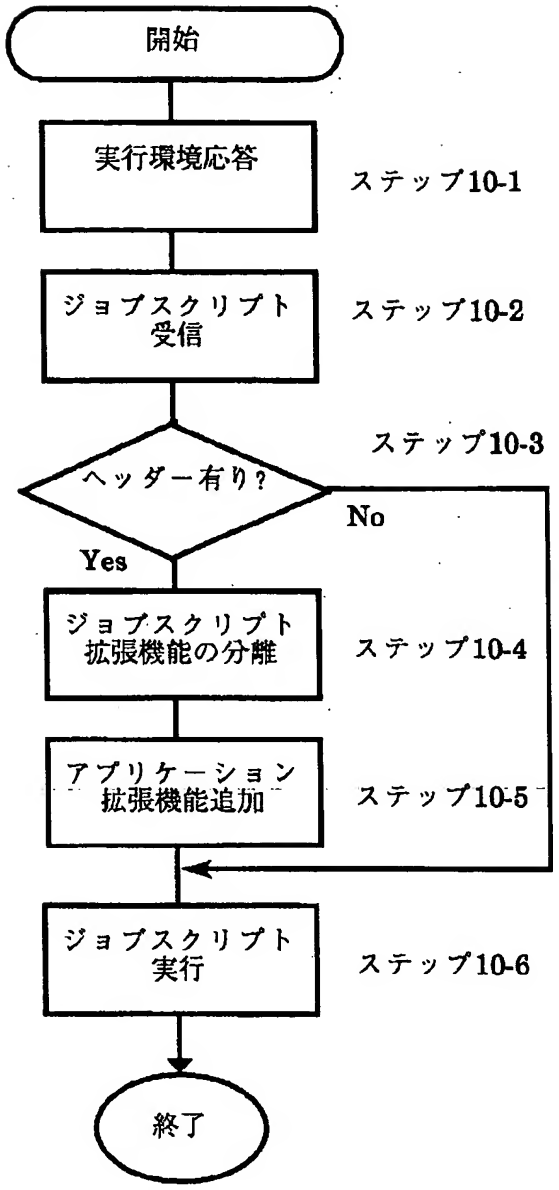

図10: マシン12側の処理の流れ

【図１７】

	機能1	機能2	機能3	機能4	機能5	
Application 1	1	1	1	1	0	
Application 2	0	1	1	1	1	
Application 3	1	0	1	1	1	
Application 4	1	1	1	0	1	

1: 当該機能有り
0: 当該機能無し

図17: アプリケーション名/機能名対応表

【図１８】

181 ドロー系ソフト	機能1	機能2	機能3
Application 1	1	1	1
Application 2	0	1	1
Application 3	1	0	1

182 ペイント系ソフト	機能4	機能5	機能6
Application 4	1	1	1
Application 5	0	1	1
Application 6	1	0	1

図18: カテゴリー別 アプリケーション名/機能名対応表

【図２０】

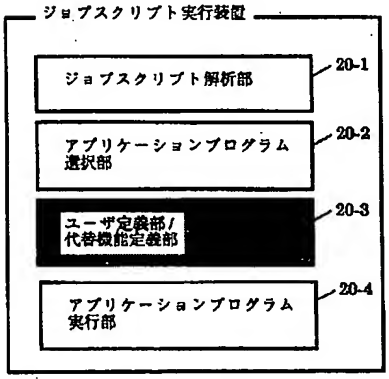

図20: ジョブスクリプト実行装置構成図

【図２３】

欠如機能名	機能の組合せ
Application 2機能1	アプリ3機能2＋アプリ4機能5

図23: ユーザ定義機能表(2)

【図２６】

欠如機能名	機能の組合せ
機能1	機能3
機能2	機能2＋機能5＋機能6
機能3	機能1＋機能3

図26: 代替機能表(1)

【図３０】

```
tell application "Microsoft Exel"
    Activeate "テストファイル1"
    Select Range ("R1B1:R2B2")
    Set Font of Selection to "明朝"
    Set Size of Selection to 12
end tell
```

図30: 実行すべきジョブスクリプトA

【図１９】

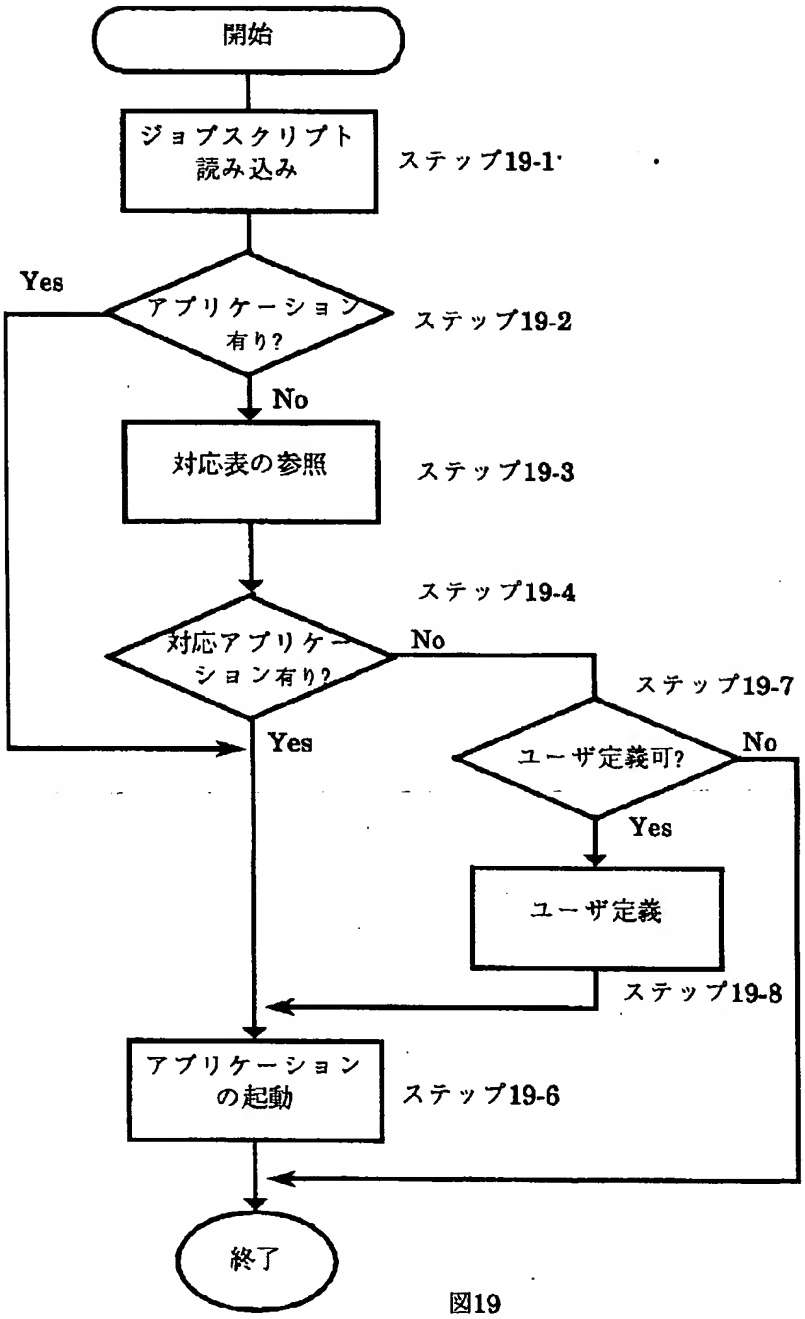

図19

【図３１】

syntax	Script Language
tell application	Apple Script
sub	Visual basic
open program	structured script
:	:

図31: 構文/スクリプト言語対応表

【図２７】

欠如機能名	機能の組合せ
Application 2機能1	アプリ3機能2＋アプリ4機能5

図27: 代替機能表(2)

【図３３】

```
Sub dummy1()
    Sheets("テストファイル1").Select
    Range("A1:B2").Select
    Selection.Font.Name = "明朝"
    Selection.Font.Size = "12"
End Sub
```

図33: 変換後のVisualBasic のジョブスクリプトB

【図２１】

	機能1	機能2	機能3	機能4	機能5	
Application 1	1	1	1	1	0	
Application 2	2	1	1	1	1	
Application 3	1	2	1	1	1	
Application 4	1	1	1	0	1	
						2: ユーザ定義機能有り 1: 当該機能有り 0: 当該機能無し

図21: アプリケーション名/機能名対応表

【図２５】

	機能1	機能2	機能3	機能4	機能5	
Application 1	1	1	1	1	0	
Application 2	2	1	1	1	1	
Application 3	1	2	1	1	1	
Application 4	1	1	1	0	1	
						2: ユーザ定義機能/代替定義機能有り 1: 当該機能有り 0: 当該機能無し

図25: アプリケーション名/機能名対応表

【図２８】

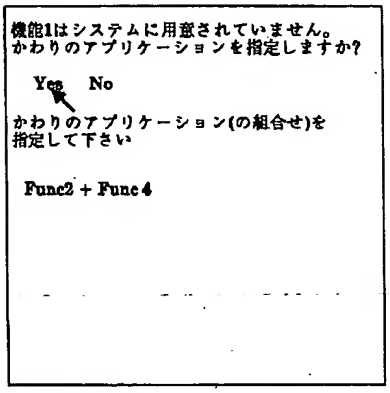

図28: ユーザ定義機能のユーザインタフェースの例

【図２９】

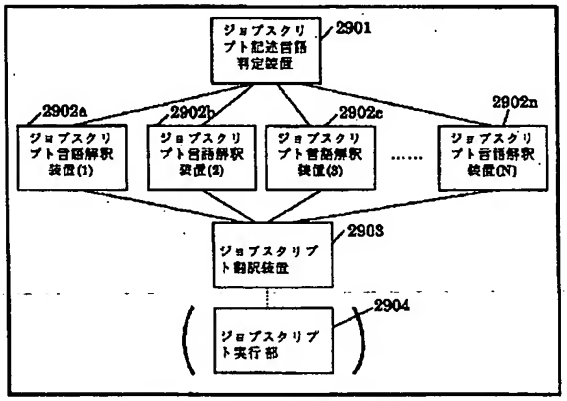

図29: ジョブスクリプト実行装置構成図

【図３２】

解釈結果	命令文
フォント名の指定	Selection.Font.Name
フォントサイズの指定	Selection.Font.Size
:	:

図32: 意味/命令文対応表

【図３５】

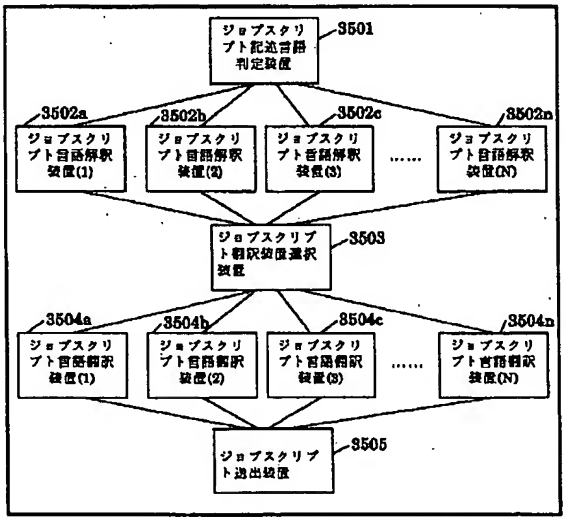

図35: ジョブスクリプト実行装置構成図

【図２４】

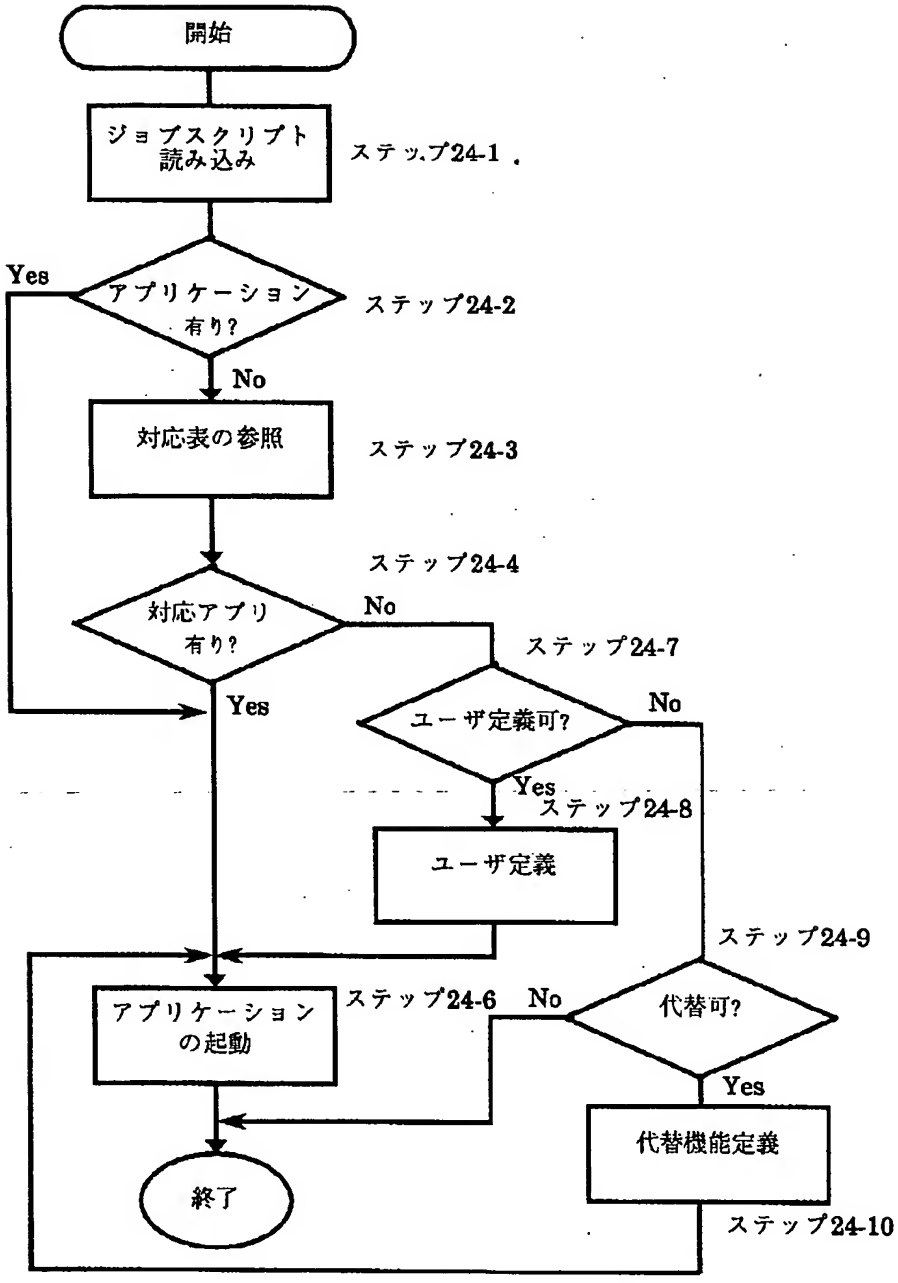
開始
ジョブスクリプト読み込み
ステップ24-1
アプリケーション有り?
Yes
No
ステップ24-2
対応表の参照
ステップ24-3
ステップ24-4
対応アプリ有り?
No
Yes
ステップ24-7
ユーザ定義可?
No
Yes
ステップ24-8
ユーザ定義
ステップ24-9
アプリケーションの起動
ステップ24-6
No
代替可?
Yes
代替機能定義
ステップ24-10
終了

図24

【図３４】

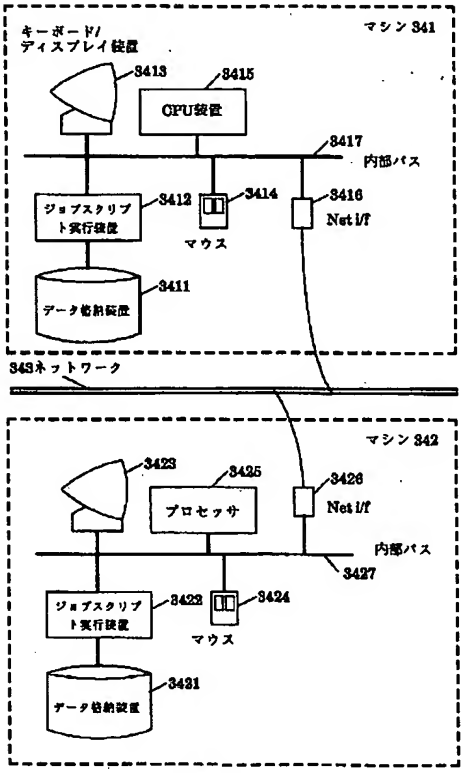

図34

【図３６】

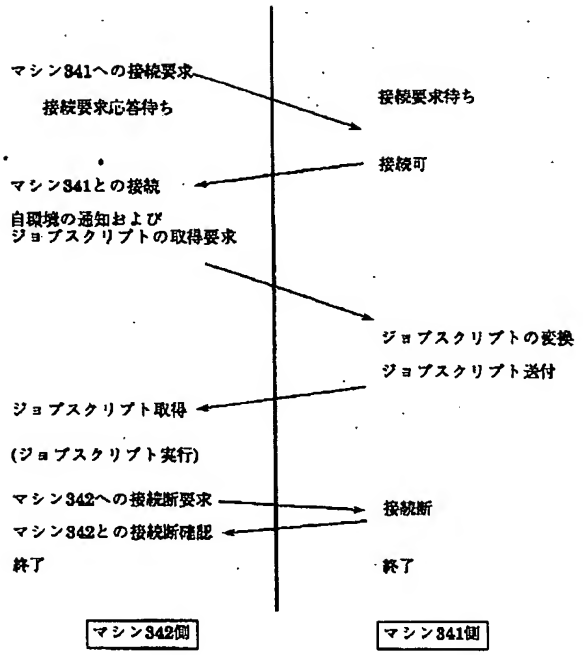

図36: マシン341とマシン342との通信プロトコル

フロントページの続き

(51) Int. Cl.[6]	識別記号	庁内整理番号	FI		技術表示箇所
G06F 9/06	540		G06F 9/06	420J	
G06T 11/00			15/72	G	